PROVA 2

MAC110 INTRODUÇÃO À COMPUTAÇÃO

1o. SEMESTRE DE 2017

Nome:

Número USP:

Instruções:

(*1*) Esta prova é individual.

(*2*) Não destaque as folhas deste caderno.

(*3*) A prova consiste de 6 questões (contando a Questão 0 nesta página).

(*4*) As respostas devem estar nos locais indicados.

(*5*) Não é permitido o uso de aparelhos eletrônicos de qualquer natureza.

(*6*) Não é permitido o uso de folhas avulsas para rascunho.

(*7*) Não é necessário apagar seus rascunhos.

(*8*) Não é permitido consultar qualquer material ou consultar colegas.

Assinatura:

Sua assinatura acima atesta a autenticidade e originalidade de seu trabalho e que você se compromete a seguir o código de ética da USP em suas atividades acadêmicas, incluindo esta prova.

Boa sorte!

Q	0	1	2	3	4	5	**Total**
Nota							

Q0. [5 pontos] Leia o conteúdo desta página e preencha os itens requisitados. Assine acima, e atente ao significado de sua assinatura.

Data: 2017/6/29, 7:21pm

Q1. [16 pontos] Considere o programa

```
public class FunctionQ {
    public static int[] f(int[] x, int a) {
        for (int i = 0; i < x.length; i++) x[i] *= a;
        a = x[2];
        return x;
    }
    public static void main(String[] args) {
        int[] x = {1, 2, 3};
        int a = 4;
        int[] y = f(x, a);
        y[0] = 0;
        StdOut.println(a); StdOut.println(x[0]);
        StdOut.println(y[0]); StdOut.println(x[1]);
    }
}
```

Este programa tem quatro linhas de saída. Diga quais são as saídas abaixo:

(*i*) Primeira linha: *Resp.*: `4`

(*ii*) Segunda linha: *Resp.*: `0`

(*iii*) Terceira linha: *Resp.*: `0`

(*iv*) Quarta linha: *Resp.*: `8`

Q2. [25 pontos] Considere as seguintes duas funções recursivas.

```
public static void f(int n) {
    for (int i = 0; i < n; i++) f(i);
    StdOut.print(n + " ");
}

public static int g(int n) {
    if (n % 2 == 0) return n/10;
    return g(g(n/10));
}
```

(*i*) O que é impresso quando é executada a chamada `f(0)`?

Resp.: `0␣`

(*ii*) O que é impresso quando é executada a chamada `f(1)`?

Resp.: `0␣1␣`

(*iii*) O que é impresso quando é executada a chamada `f(2)`?

Resp.: `0␣0␣1␣2␣`

(*iv*) O que é impresso quando é executada a chamada `f(3)`?

Resp.: `0␣0␣1␣0␣0␣1␣2␣3␣`

(*v*) Qual é o valor de `g(7500231)`? *Resp.*: `75`

(*vi*) Qual é o valor de `g(1742075)`? *Resp.*: `17`

Q3. [24 pontos] Considere o seguinte programa:

```
public class Mystery {
    public static int mystery(int a, int b) {
        if (b == 0) return 1;
        if (b % 2 == 0) return mystery(a * a, b/2);
        return a * mystery(a * a, b/2);
    }
    public static void main(String[] args) {
        for (int a = 1; a <= 3; a++) {
            for (int b = 0; b <= 3; b++)
                StdOut.print(mystery(a, b) + " ");
            StdOut.println();
        }
    }
}
```

(*i*) O que é impresso pelo programa acima quando executado?

Resposta:

```
1 1 1 1
1 2 4 8
1 3 9 27
```

(*ii*) Quanto é `mystery(2,13)`? *Resp.*: `2^13 = 8192`

Q4. [20 pontos] Os itens abaixo especificam uma quantidade que depende de um parâmetro N. Diga se esta quantidade tem ordem de grandeza *logarítmica*, *linear*, *quadrática*, *cúbica*, ou *exponencial* em N. [*Obs.* São exemplos de quantidades exponenciais: 2^N, $(3/2)^N$, $3^N/N^2$, etc.]

(*i*) Memória usada pela declaração e inicialização

```
int[][] a = new int[N][N];
```

Resp.: `quadrático`

(*ii*) Tempo usado pela declaração e inicialização

```
int[][] a = new int[N][N];
```

Resp.: `quadrático`

(*iii*) Tempo gasto pelo trecho de código

```
int t = 0;
while (N > 0) { t++; N /= 2; }
```

Resp.: `logarítmico`

(*iv*) Comprimento do string devolvido pela chamada `f(N)`, onde `f()` é a função abaixo:

```
public static String f(int N) {
    if (N == 0) return "";
    if (N == 1) return "x";
    return f(N - 1) + f(N - 2);
}
```

Resp.: `exponencial`

(*v*) Tempo usado pela chamada `ff(N)`, onde a função `ff()` é a função abaixo:

```
public static int ff(int N) {
    int[] v = new int[N + 2];
    v[0] = 0; v[1] = 1;
    for (int i = 2; i <= N; i++) v[i] = v[i - 1] + v[i - 2];
    return v[N];
}
```

Resp.: `linear`

Q5. [20 pontos] Um programador observa o seguinte comportamento de um programa:

```
N       Tempo
2000    1.0 minuto
4000    2.8 minutos
8000    7.9 minutos
16000   22.8 minutos
```

Na tabela acima, N é o número de objetos que o programa recebe como entrada, e o tempo indicado é quanto o programa demora para produzir a saída. Suponha que o programa demora tempo aproximadamente aN^b para certas constantes a e b (a e b não necessariamente inteiros). Suponha que o programador irá executar este programa com $N = 256000$. Marque abaixo sua estimativa para quanto tempo o programa vai demorar:

(a) meia hora (b) duas horas **(c) um dia** (d) um mês (e) dois meses

O programador decide então que ele precisa melhorar o algoritmo implementado em seu programa. Depois de estudar os algoritmos disponíveis e implementar um algoritmo melhor, ele observa o seguinte comportamento:

```
N       Tempo
2000    1.0 minuto
4000    2.0 minutos
8000    3.9 minutos
16000   8.1 minutos
```

Novamente, suponha que este programa demora tempo aproximadamente aN^b para certas constantes a e b (a e b não necessariamente inteiros). Marque abaixo sua estimativa para quanto tempo este novo programa vai demorar para $N = 256000$:

(a) meia hora **(b) duas horas** (c) um dia (d) um mês (e) dois meses